МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент
Технический директор
ООО «Инновации детям»
_____ И.И. Григорьев
“___”_____ 2020 г

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой,
д.ф.-м.н., профессор

_____Л.Б.Соколинский
“___”_____ 2020 г

Разработка веб-приложения для создания персонажа DnD игр

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.03.02.2020.308-418.ВКР

Научный руководитель,
ст. преподаватель кафедры СП
_____П.Г. Верман

Автор работы,
студент группы КЭ-401
_____Р.О. Касумов

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___» _____ 2020 г.

Челябинск 2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
03.02.2020

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-401
Касумову Роману Олеговичу,
обучающемуся по направлению
02.03.02 «Фундаментальная информатика и информационные технологии»

1. Тема работы (утверждена приказом ректора от 24.04.2020 № 627)
Разработка веб-приложения для создания персонажа DnD игр
2. Срок сдачи студентом законченной работы: 08.06.2020 г.
3. Исходные данные к работе
3.1. React. [Электронный ресурс] URL:https://reactjs.org (дата обращения 28.02.2020).
3.2. Нильсен Я., Лоранжер Х. Web-дизайн: удобство использования Web-сайтов. - Москва [и др.]: Вильямс, 2009. - 376 с.
3.3. Dangeons&Dragons. [Электронный ресурс] URL: https://dnd.wizards.com/ (дата обращения 08.04.2020).
4. Перечень подлежащих разработке вопросов
4.1. Выполнить анализ предметной области и произвести обзор существующих решений.
4.2. Спроектировать веб-приложение.
4.3. Реализовать веб-приложение.
4.4. Провести тестирование.
5. Дата выдачи задания: 03.02.2020 г.

Научный руководитель П.Г. Верман

Задание принял к исполнению Р.О. Касумов

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 4
АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6
1.1.Что такое DnD 6
1.2.Отличия редакций DnD 7
1.3.Персонаж DnD 9
1.3.1. Расы 9
1.3.2. Классы 11
1.3.3. Личность и предыстория 11
1.3.4. Снаряжение 12
1.4.Обзор существующих решений 13
1.4.1. D&D Beyond 13
1.4.2. Character builder 15
1.4.3. Roll20 15
2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ 18
2.1.Требования к системе 18
2.2.Варианты использования 18
2.3.Архитектура веб-приложения 20
2.4.Схема базы данных сервера системы 22
2.5.Интерфейс 22
3. РЕАЛИЗАЦИЯ СИСТЕМЫ 26
3.1.Инструменты реализации 26
3.2.Реализация сервера 26
3.3.Маршрутизация 28
3.4.Реализация клиента 29
3.5.Пользовательский интерфейс 32
4. ТЕСТИРОВАНИЕ СИСТЕМЫ 37
ЗАКЛЮЧЕНИЕ 41

ВВЕДЕНИЕ

Актуальность

В настоящее время веб-сервисы являются главным средством решения повседневных или нестандартных задач. Покупка вещей, просмотр новостей, общение – одни из многих задач, для которых мы используем веб-сервисы. Это помогает избежать ненужной траты времени, необходимости посетить магазин для того, чтобы посмотреть и оценить товар. Настольные игры не являются исключением. Для многих видов настольных игр требуется создание персонажа (описание характеристик) на специальном листе. Для того, чтобы сделать процесс заполнения удобным и простым, существуют специализированные веб-ресурсы. Такие ресурсы могут упростить знакомство с игрой для новичков.

Существует множество видов настольных игр, но мы будет говорить об одной из культовых настольных игр – Dungeons & Dragons[4]. Это очень комплексная игра, в которой принимают участие несколько игроков и мастер. Одна из больших проблем игроков – создание персонажа. Все начинается с заполнения листа персонажа. Он обладает множеством полей: имя, раса, класс, возраст, распределение очков характеристик, снаряжение и т.д. Так как игра чаще всего состоит из нескольких встреч, некоторые игроки могут прекратить играть и на замену им придут новые, не знающие, где сейчас находиться группа, какой уровень у других персонажей, сколько очков он может потратить на распределение характеристик. Для того, чтобы структурировать и сделать понятным процесс заполнения листа персонажа, чтобы не искать и не разбираться в информации о соответствии способностей персонажа его игровому уровню, можно разработать веб-приложение.

Цели и задачи

Целью данной работы является разработка веб-приложения для создания персонажа dnd игр, в котором можно заполнить листы персонажей

настольной игры Dungeons & Dragons для дальнейшей игры с друзьями оффлайн.

Для выполнения поставленной цели, необходимо решить следующие задачи.

1. Выполнить анализ предметной области и произвести обзор существующих решений.

2. Спроектировать веб-приложение.

3. Реализовать веб-приложение.

4. Протестировать веб-приложение.

Также разрабатываемое приложение должно удовлетворять следующим условиям.

1. Иметь удобный и понятный дизайн.

2. Предоставлять возможность хранения созданный персонажей.

3. Предоставлять возможность получить информацию о персонаже в формате pdf.

Структура работы

Работа состоит из введения, четырех глав, заключения, библиографии и приложения. Объем работы составляет 43 страниц, объем библиографии составляет 16 источников.

Содержание работы

Первая глава «Анализ предметной области» содержит постановку задачи и обзор аналогичных проектов.

Вторая глава «Проектирование системы» содержит описание и анализ требований к веб-приложению и серверу, описание архитектур приложения и сервера, а также описание схем баз данных приложения и сервера.

Третья глава «Реализация системы» описывает подробности реализации сервера и клиента для веб-браузера.

Четвертая глава «Тестирование системы» посвящена результатам тестирования сервера и клиента для веб-браузера.

В заключении описываются основные результаты, полученные при выполнении дипломной работы.

АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1.Что такое DnD

Dungeons & Dragons – это настольная ролевая игра, в которой игроки берут на себя роли героев, образующих группу, для путешествий в опасные приключения. Настольная ролевая игра – вид ролевой игры, в которой участники устно описывают действия своих персонажей, опираясь на их особенности, и в котором успех действий зависит от игровой системы. Пройти по сюжету им помогает Игровой мастер, который решает, с какими приключениями столкнуться игровые персонажи, какое вознаграждение они получат за выполнение своего квеста или последствия неудачи. И книга игрока[5], в которой находиться вся необходимая информация для создания персонажа. *Квест* – это задание, выполнив которое, персонаж игрока или его группа получает награду.

Игрок принимает все решения за своего персонажа, начиная с его способностей и заканчивая оружием и снаряжением. Также игрок может определить личность своего персонажа. Например, он может быть благородным рыцарем, который отправился на сражения

Мастер игры контролирует исследуемый игроками мир. Его работа обеспечить игроков честными и интересными заданиями. От начала и до конца он управляет персонажами, которые не отыгрываются игроками. Мастер должен сделать все возможное, чтобы игроки ощущали реальность происходящего. Только совершая действия во вселенной, которая кажется им правдоподобной, они ощущают вес и значение своих действий, в полной мере принимают их последствия. Правила проведения, механики и правила игры могут различаться в различны редакциях.

1.2.Отличия редакций DnD

Правила Dungeons & Dragons прошли через несколько редакций. Оригинальные правила были опубликованы в 1974 и были дополнены в следующие два года. В 1977 TSR выпустила две новые версии: Advanced DnD (AD&D) и DnD. Dungeons & Dragons был облегченной версией игры, а Advanced была более сложной версией. в 1977-1979 были выпущены три кнупные книги правил, называемые «Основным руководством». Вторая редакция AD&D была сильно переделана. Были удалены классы «убийца» и «монах». «Демоны» и «дьяволы» были переименованы, а затем полностью удалены из игры.

В 2008 была выпущена 4-я редакция, главными отличиями которой являются: увеличение количества уровней класса с 20 до 30; значительное усиление влияния выбранной расы на возможности персонажа; изменение в списке доступных для игры рас; введение специализаций для воинов (быстрый и мощный стиль боя; изменение существующей системы мировоззрения; новый игровой мир.

В 2014 была выпущена 5-я редакция, которая является последней, на данный момент. Данной редакции я уделю больше внимания, так как на основе нее будет выполняться данная работа. Выделим 5 главных особенностей пятой редакции.

Фокус игры

Основной фокус игры по-прежнему сосредоточен не только на сражениях, но и подготовке к ним, однако странствовать по длительным приключениям, раскрывать характеры персонажей и их личные истории теперь стало намного удобнее. Персонажи теперь должны иметь идеалы, недостатки и связи, что углубляет нарратив.

Упрощение системы взаимодействия

Было затронуто несколько аспектов боя и небоевых встреч и событий. Одним из таких упрощений стало введение механики преимуществ и помех. Вместо подсчета постоянных модификаторов к броску теперь

применяется механизм преимуществ и помех, позволяющие просто решать вопросы как в бою, так и вне боя. Это делает механику более простой и понятной.

Это так же повлияло на продолжительность игр, что многие игроки сочли за положительные изменения.

Система развития персонажа

Изменился порядок прокачки персонажа. Особые качества стали опциональным элементом, могут браться и использоваться по желанию игрока, вместо поднятия атрибутов с уровнем. Уменьшилась скорость роста характеристик в рамках прогрессии персонажа. Вместо тренировки навыков было решено внедрить механику владения определенными сферами, которая касается как доспехов или оружия, так и различных знаний, и обращения с небоевыми предметами.

Теперь балансирование точек событий стало более легкой задачей для мастера, легкие столкновения теперь не будут казаться такими простыми, встретить не убиваемого противника так же будет достаточно тяжело.

Магия

Теперь практически каждый маг может использовать заклинания нулевого круга сколько угодно и не становиться бесполезным для группы, когда заканчиваются его слоты заклинаний. показывающая сложность заклинания. *Нулевой круг* - мера сложности овладения заклинанием (чем выше круг заклинания, тем больший уровень персонажа требуется для овладения этим заклинанием. Лимит по количеству используемых заклинаний в день привязаны не к уровню персонажа, а к его уровню в определенном классе. При этом подготовка заклинаний заранее позволяет не фиксировать их в ячейках, а заготавливать список заклинаний, и использовать любое из подготовленных, пока что хватает слотов, что расширяет тактические возможности заклинателей.

Можно сказать, что магия стала многогранной и разнообразной, но также уменьшилось количество возможных способов использования магии для чрезмерного повышения способностей персонажей.

Вывод

Из изученной информации, можно сделать вывод, что пятая редакция одна из самых дружелюбных к игрокам. Начать можно практически без длительной подготовки. Книга игрока теперь обладает советами о том, как следует взаимодействовать за столом, и в целом направлена на людей, которые так же не играли в настольные ролевые игры.

1.3. Персонаж DnD

Персонаж является сочетанием игровой стилистики, ролевых зацепок и воображения. Для начала игрок должен выбрать расу и класс. Каждый персонаж принадлежит к одной из множества разумных гуманоидных рас мира D&D.

Игрок выбирает расу и класс, создаёт личность, внешность и предысторию своего персонажа.

Каждый персонаж принадлежит к одной из множества разумных рас мира D&D. Некоторые расы также могут иметь подрасы. Также игроку необходимо выбрать класс, от которого будут зависеть таланты и призвание персонажа, которые он получит. Так же персонаж получает классовые умения - возможности, которые отличают вашего персонажа от представителей других классов. Уровень начинается с первого и повышается на протяжении всей игры.

1.3.1. Расы

Наиболее распространенными расами игровых персонажей являются дварфы, эльфы, полурослики и люди. Реже встречаются драконорождённые, полуэльфы, полуорки, гномы и тифлинги. Следующие характери-

стики встречаются в большинстве рас: возраст, мировоззрение, размер, скорость, языки, разновидности расы.

Каждая раса увеличивает одно или несколько значений характеристик. При выборе расы Эльф, персонаж получает дополнительные две единицы к показателю Ловкость.

Возраст помогает игроку судить, может ли его персонаж считаться старым или молодым по меркам его расы. Также это может служить обоснованием основных показателей персонажа (сила, ловкость, мудрость и т.д.)

Большинство рас склоняются к определенному мировоззрению, но это никак не ограничивает персонажа. Мировоззрение не влияет на прямую на характеристики персонажа, но может послужить обоснованию его характера.

Размер существа может влиять на количество существ, которое может принять участие в бою. Пять существ большого размера столпятся вокруг одного Среднего, больше ни для кого места не останется и наоборот

Скорость определяет, количество футов на которые персонаж может переместится в рамках одного хода.

В зависимости от расы, персонаж может говорить, читать и писать на определенных языках.

Расы могут иметь нескольких представителей. Представители сочетают в себе качества основной расы с особенностями, присущими только этой разновидности. Например, представитель Высший эльф добавляет к базовому значению интеллекта 1 очко.

В качестве примере, рассмотрим расу Эльф.

Раса *«Эльфы»* – волшебный народ, живущий в мире, но не являющийся его частью. Живут в глубинах древних лесов или серебряных жилищах. Любят природу, магию, музыку и поэзию. *Подрасы*: Лесной эльф, Темный эльф, Высший эльф. Пример некоторых характеристик на основе бонуса расы.

1. *Обострённые чувства*: владение навыком «Внимательность».

2. *Наследие фей*: совершение с преимуществом бросков от очарования, и вас невозможно магически усыпить.

1.3.2. Классы

Класс дает целый ряд особых умений, таких как мастерство воина владении оружием и доспехами, и заклинаниями волшебника. На низких уровнях класс дает только два или три умения, при получении новый уровней игрок получит новые, и старые умения улучшатся.

Иногда персонажи могут развиваться сразу в нескольких классах. Плут может поменять жизненный путь и дать присягу паладина. В игре существуют 12 классов: Бард, Варвар, Волшебник, друид, Жрец, Колдун, Монах, Паладин, Плут, Следопыт, Чародей.

Класс *Варвар* – параметры.

1. *Доспехи: Легкие доспехи; средние доспехи; щиты.*
2. *Оружие: Простое оружие; воинское оружие.*
3. *Инструменты: нет.*
4. *Спасброски: Сила; Телосложение.*
5. *Доступные навыки: Внимательность; Выживание; Запугивание; Природа; Уход за животными.*
6. *Снаряжение: Секира или любое воинское рукопашное оружие; Два ручных топора или любое простое оружие; Набор путешественника и четыре метательных топора.*
7. *Ярость: Совершение спасбросков Силы и проверки с преимуществом; Сопротивление дробящему, колющему и рубящему урону.*

1.3.3. Личность и предыстория

Персонажи характеризуются не только расой и классом. Они – личности, с собственной историей, связями и возможностями, выходящие за

определения классов и рас. Описание расы персонажа включает примеры имен для представителей этой расы.

Можно определить рост и вес своего персонажа, используя информацию в описании расы, которую выбрал игрок. Игрок выбирает возраст, цвет волос, глаз и кожи своего персонажа. От расы будет зависеть, на каких языках персонаж сможет говорить по умолчанию, а предыстория может дать еще несколько языков на выбор.

Также нужно определить две черты характера своего персонажа. Черты могут рассказывать о достижениях персонажа, его пристрастиях или страхах, его манерах и самооценки. Можно придумать черты характера, посмотрев на максимальную и минимальную характеристику своего персонажа, и придумав по черте, связанной с ними.

1.3.4. Снаряжение

Когда игрок создает персонажа, он получает снаряжение, состав которого определяется классом и предысторией, которые он выбрал. В качестве альтернативы, игрок может получить монеты и выбрать снаряжение самостоятельно. Любой персонаж может носить доспехи и оружие. Выбранный класс дает владение некоторыми видами доспехов. Если персонаж носит доспехи, которыми не владеет, он будет совершать с помехой все проверки характеристик, спасброски и броски атаки.

Доспехи

Легкие доспехи, изготовленные из лёгких и тонких материалов, предпочитают ловкие искатели приключений, поскольку те предоставляют защиту, и при этом не ограничивают подвижность. Если персонаж носит лёгкий доспех, при определении Класса Доспеха добавляете модификатор Ловкости к базовому числу, предоставленному доспехом. Средние доспехи предлагают лучшую защиту, чем лёгкие, но немного ограничивают перемещение. Тяжёлые доспехи предоставляют лучшую защиту. Тяжёлый до-

спех не позволяет добавлять к Классу Доспеха модификатор Ловкости, но и не даёт штраф, если модификатор ловкости отрицательный.

Оружие

Раса, класс и черты дают владение конкретными видами и целыми категориями оружия. Категорий две: простое оружие и воинское оружие. Простым оружием могут пользоваться многие. Воинское оружие включает мечи, топоры и древковое оружие, требующее для эффективного использования особых тренировок. Владение оружием позволяет вам добавлять бонус мастерства к броскам атаки всех атак, совершённых этим оружием. Если у оружия есть свойство «метательное», персонаж может совершать им дальнобойные атаки, метая его. Если это рукопашное оружие, персонаж использует для бросков атаки и урона тот же модификатор характеристики, что и при совершении рукопашной атаки этим оружием.

Инструменты

Инструменты помогают создавать или чинить вещи, подделывать документы и вскрывать замки. Владение инструментом позволяет добавлять бонус мастерства к проверкам характеристик, сделанным при использовании этих инструментов. Использование инструментов не привязано к какой-то одной характеристике, так как владение инструментом отражает широкие познания в его использовании.

Например, *воровские инструменты* – это набор инструментов, в который входят: небольшой напильник, набор отмычек, небольшое зеркальце на длинной ручке, ножницы и пара щипчиков. Владение данными инструментами позволяет добавлять бонус мастерства ко всем проверкам характеристик, сделанным для отключения ловушек и взлома замков.

1.4.Обзор существующих решений

1.4.1. D&D Beyond

«D&D Beyond» портал, предоставляющий базу знаний, инструменты для создания своего персонажа, магазин, гайды для новичков и многое

другое. Builder предоставляет 3 шаблона для создания персонажа. *Standart* позволяет создать персонажа шаг за шагом, как указывают авторы, этот вариант хорошо подходит новичкам. *Quick build* предлагает создать персонажа первого уровня, используя рекомендованные стартовые настройки. *Randomize* позволяет создать персонажа с любым уровнем и случайными характеристиками из выбранных настроек. Интерфейс представлен на рисунке 1. К достоинствам можно отнести.

1. Подробное описание умений и предметов.
2. Красивый дизайн.
3. Понятный интерфейс.
4. Дружелюбность к новичкам.
5. Несколько вариаций создания персонажа.
6. Удобный UI дизайн.

К недостаткам можно отнести.

1. Наличие большого количества опций и настроек, которые могут запутать пользователя.
2. Отсутствие русского языка.

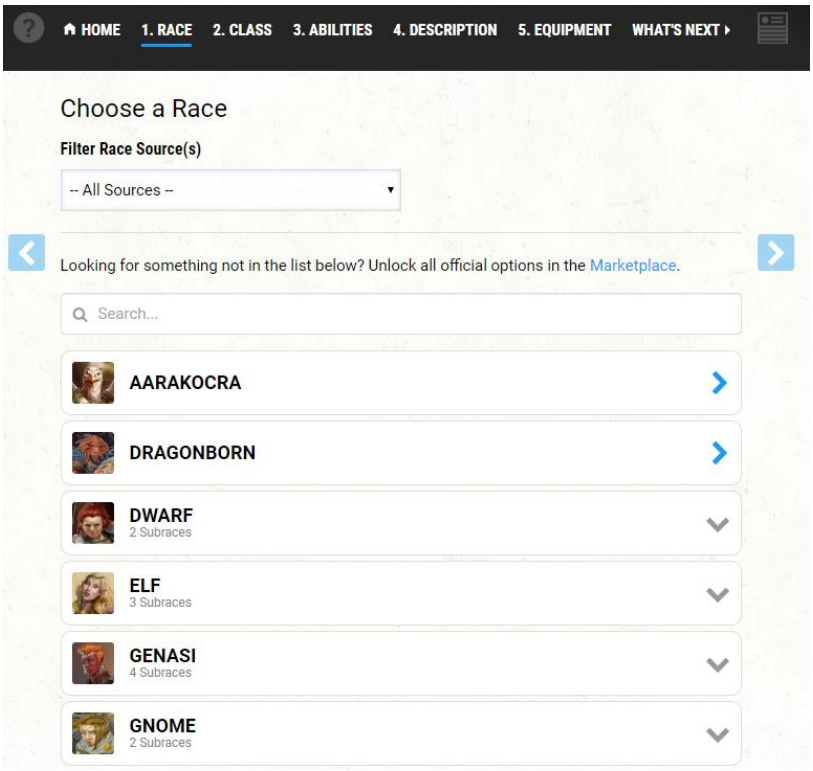

Рис. 1. Интерфейс генератора персонажа сайта D&D Beyond

1.4.2. Character builder

Сайт предоставляет возможность создать своего персонажа. При входе на сайт, пользователю сразу предлагают заполнить данные о персонаже. Для этого нужно поэтапно пройти по всем страницам и в конце получить готовый лист. Интерфейс представлен на рисунке 2.

К плюсам можно отнести.

1. Подробное описание умений и предметов.
2. Сохранение уже созданных персонажей.
3. Демонстрация уже выбранных характеристик.

К минусам можно отнести.

1. Отсутствие информации о редакции, по которой создается персонаж.
2. Простой дизайн.
3. Неудобная навигация.
4. Отсутствие русского языка.
5. Отсутствие расширенных настроек.

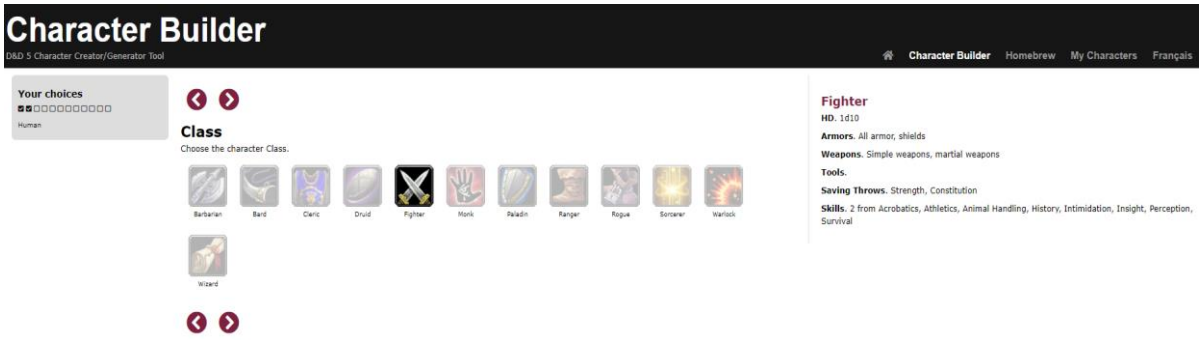

Рис. 2. Интерфейс сайта Character Builder

1.4.3. Roll20

Из всех представленных примеров этот портал имеет не только инструменты для создания персонажа, но и дает возможность провести полноценную игру прямо на портале. Основное внимание уделим генератору персонажа. Пользователю предоставляют несколько вариантов для создания персонажа: *charactermancer, character sheet.* Первый вариант предпо-

чтителен для новичков, т.к. предоставляет пошаговое создание персонажа, с подробным описанием каждого этапа. Второй вариант более популярный, он предоставляет весь лист персонажа целиком. Интерфейс представлен на рисунке 3.

К плюсам можно отнести.

1. Удобный и понятный интерфейс.
2. Множество языков.
3. Несколько шаблонов для создания персонажа.
4. Возможность сохранение своего персонажа.

К минусам можно отнести.

1. Наличие небольшого количества мест для хранения персонажей.
2. Невозможность создания персонажа, не начиная игру.
3. Невозможность скачать лист персонажа.

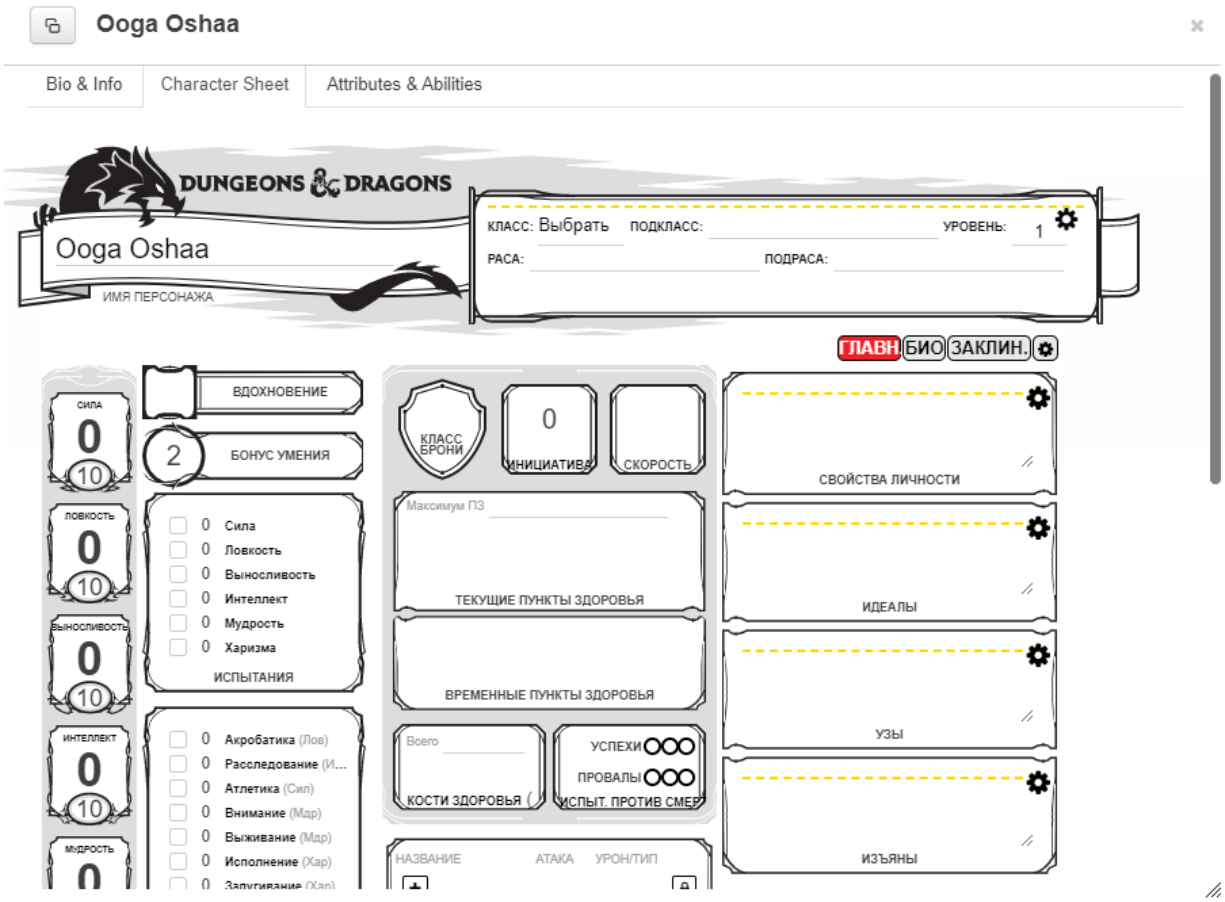

Рис. 3. Интерфейс генератора персонажа сайта Roll20

Вывод

Из рассмотренных вариантов наиболее удобными и информативными вариантами являются «Roll20» и «D&D Beyond».

Обзор аналогов показал, что в первую очередь стоит обратить внимание на интерфейс и удобство заполнения самого листа.

Пользователю следует видеть весь лист сразу, без отдельной пошаговой навигации. Также у авторизированного пользователя должна быть возможность сохранить созданный лист.

В качестве образца, был взят лист персонажа с сервиса Roll20, так как он соответствует оригинальному листу DnD пятой версии.

2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ

В результате анализа предметной области и обзора существующих решений, были сформированы следующие требования.

2.1. Требования к системе

Разрабатываемое веб-приложение для создания персонажа должно удовлетворять следующим функциональным требованиям.

1. Веб-приложение должно предоставлять пользователю возможность регистрации и авторизации.
2. Веб-приложение должно иметь удобный и понятный UI, и UX дизайн.
3. Веб-приложение должно хранить ранее созданные листы персонажей пользователя.
4. Веб-приложение должно предоставлять готовый лист в виде pdf файла.

Нефункциональные требования

В результате анализа функциональных требований и обзора аналогичных проектов, были сформулированы следующие нефункциональные требования.

1. Приложение должно быть на языке JavaScript.
2. Сервер приложения должен быть написан на языке JavaScript с использованием платформы Node.js.
3. Сервер приложения должен иметь Rest сервис.
4. Клиент должен быть написан с использованием библиотеки React.js.

2.2. Варианты использования

Диаграмма вариантов использования веб-приложения «D&D5 Hero» представлена на рисунке 4. С приложением могут взаимодействовать два вида пользователей: неавторизованный и авторизованный. Неавторизован-

ный пользователь – человек, который еще не прошел процедуру регистрации и авторизации, соответственно, человек, который прошел эти процедуры является авторизованный.

Рис. 4. Варианты использования веб-приложения.

Неавторизованный пользователь и авторизованный пользователь могут *создать новый лист* и *скачать лист персонажа*, без возможности сохранить его у себя на аккаунте, для редактирования или повторного просмотра. Чтобы скачать лист, нажать на кнопку «Скачать» на странице создания персонажа, после чего будет скачан pdf документ со всеми характеристиками персонажа.

Авторизованный пользователь может воспользоваться функцией *сохранить лист персонажа*, для этого ему нужно нажать на кнопку «Сохранить», на странице создания персонажа. Если персонаж был создан ранее, то его данные будут изменены, в другом случае будет создан новый персонаж.

Также авторизованный пользователь может *просмотреть сохраненный лист* персонажа, перейдя на специальную вкладку «Персонажи» и выбрав нужного героя.

Во время просмотра сохраненных персонажей, пользователь может *редактировать сохраненные листы*, для это нужно нажать на кнопку «Открыть», после чего произойдет перенаправления на страницу редактирования персонажа.

Неавторизованный пользователь может *зарегистрироваться* или *авторизоваться* в системе, получив возможность сохранить персонажа, чтобы в будущем отредактировать его или скачать.

2.3. Архитектура веб-приложения

На рисунке 5 изображена диаграмма компонентов веб-приложения.

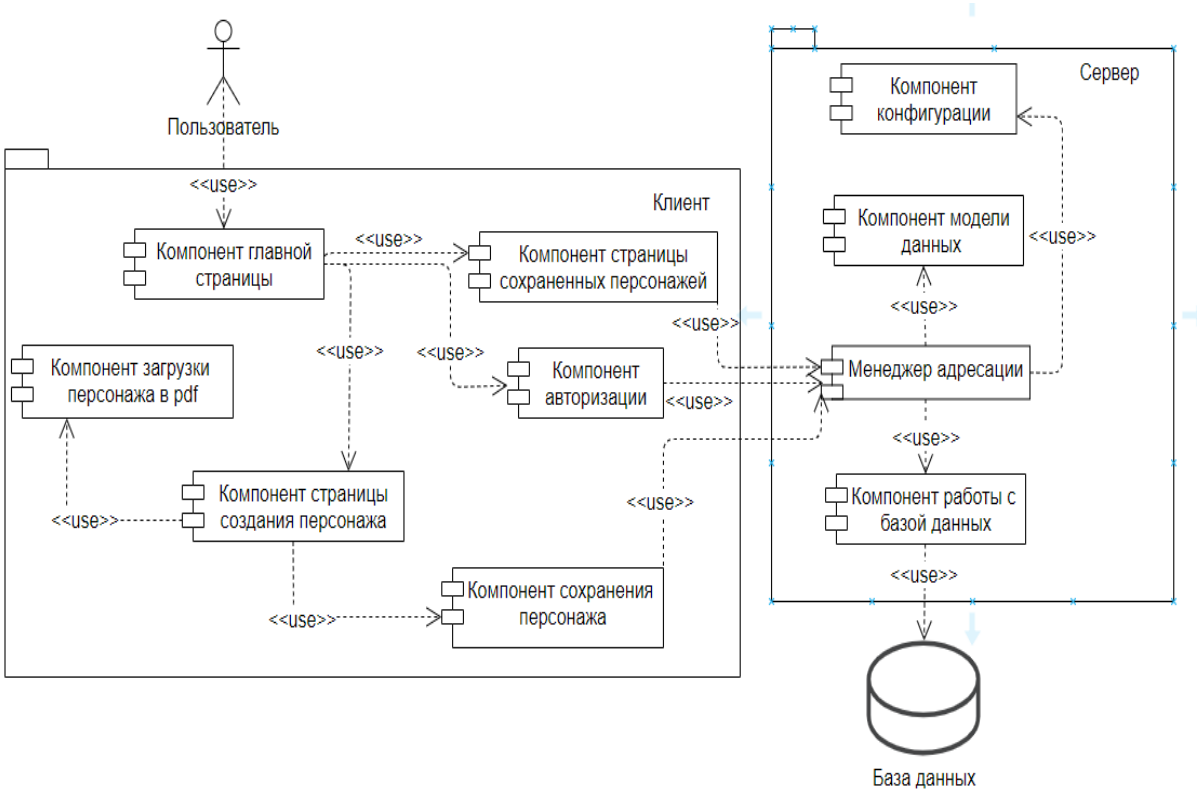

Рис. 5. Диаграмма компонентов веб-приложения.

Компонент главной страницы отрисовывает части интерфейса и в зависимости от того, авторизован пользователь или нет, предоставляет ему допустимый список ссылок и прав.

Компонент страницы сохранённых персонажей предоставляет авторизованному пользователю список сохраненных ранее персонажей, которых можно изменить или скачать в pdf формате.

Компонент авторизации предоставляет средства принятия решения о том, имеет ли клиент право на выполнение запрошенного действия.

Компонент страницы создания персонажа предоставляет пользователю интерфейс для заполнения необходимых полей и содержит логику для обработки и проверке введенных данных пользователем.

Компонент загрузки персонажа в pdf содержит логику для обработки данных введенных пользователем с последующим экспортом в pdf формате.

Компонент сохранения персонажа содержит логику, которая позволяет пользователю сохранить персонажа у себя на аккаунте, если обладает необходимыми правами доступа.

Компонент конфигурации хранит в себе статические данные, необходимые для работы других компонентов.

Компонент модели данных содержит шаблоны моделей данных, для изменений или проверки информации в базе банных.

Менеджер адресации выполняет привязку URL, по которому к серверу обратился клиент, к функции обработчику для данного URL.

Компонент работы с базой данных выполняет запросы к базе данных, позволяет отправить, запросить или изменить данные в БД.

2.4.Схема базы данных сервера системы

На рисунке 6 представлена схема базы данных сервера системы.

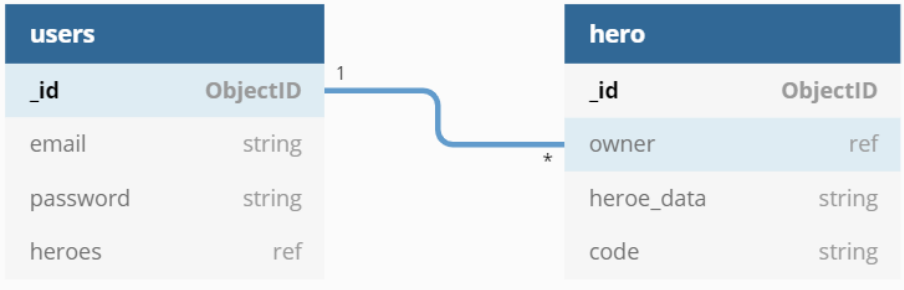

Рис. 6. Схема базы данных системы

База данных состоит из следующих таблиц.

1. *Users* - таблица, хранящая данные о пользователе и id всех связанных с ним таблиц *hero*.

2. *Hero* - таблица, хранящая данные созданного персонажа и id таблицы владельца.

2.5.Интерфейс

При входе на сайт, пользователь попадает на главную страницу. На рисунке 7 представлен макет главной страницы сайта.

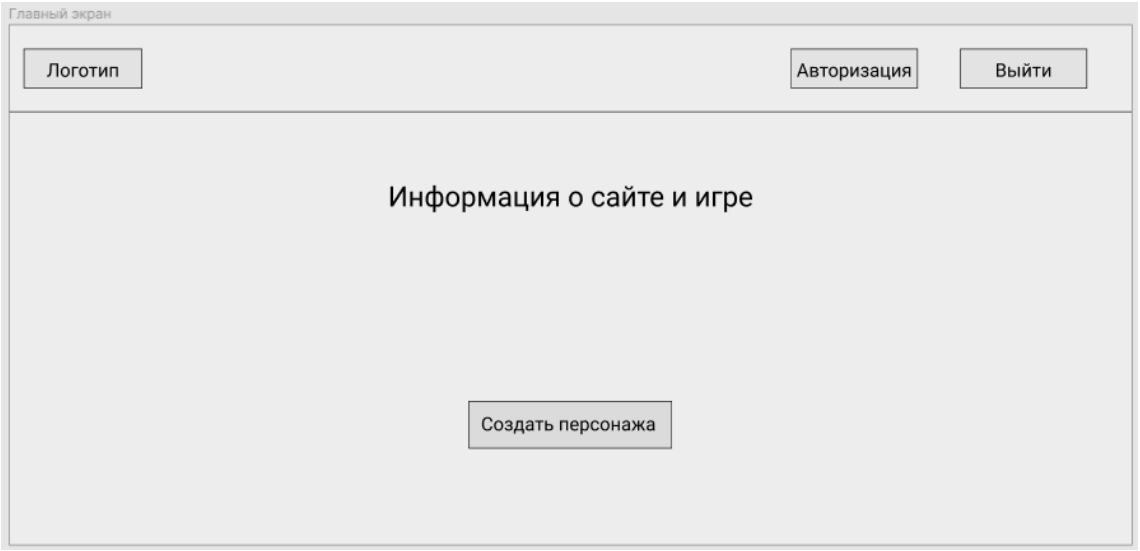

Рис.7. Макет главной страницы

На главное странице можно просмотреть общую информацию о игре, по которой будет создаваться персонаж и информацию о сайте (инструментах и функций доступных пользователю). Также пользователь может перейти на страницу для создания персонажа, нажав кнопку «Создать персонажа».

Для авторизации, пользователю нужно нажать на кнопку «Авторизация», после чего он будет перенаправлен на страницу авторизации, где требуется ввести соответствующие данные в поля «email» и «пароль» и нажать кнопку «Войти». Если у пользователя нет аккаунта, то он может пройти регистрацию по введенным данным. Макет страницы авторизации продемонстрирован на рисунке 8.

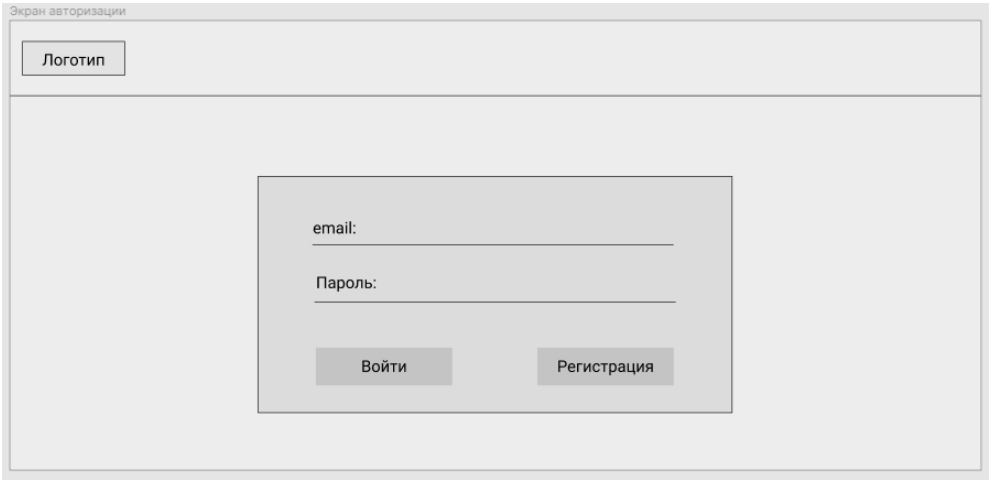

Рис. 8. Макет страницы авторизации

На странице создания персонажа, пользователю придется взаимодействовать с приложением больше всего. На странице присутствует большое количество полей разного типа, куда пользователь должен ввести данные о персонаже. Поля можно разделить на несколько типов: текстовое поле, поле с заполнением данных, поле с выбором из списка и поле с отметками, в качестве которых выступают символы галочек. На рисунке 9 представлен макет страницы для создания персонажа.

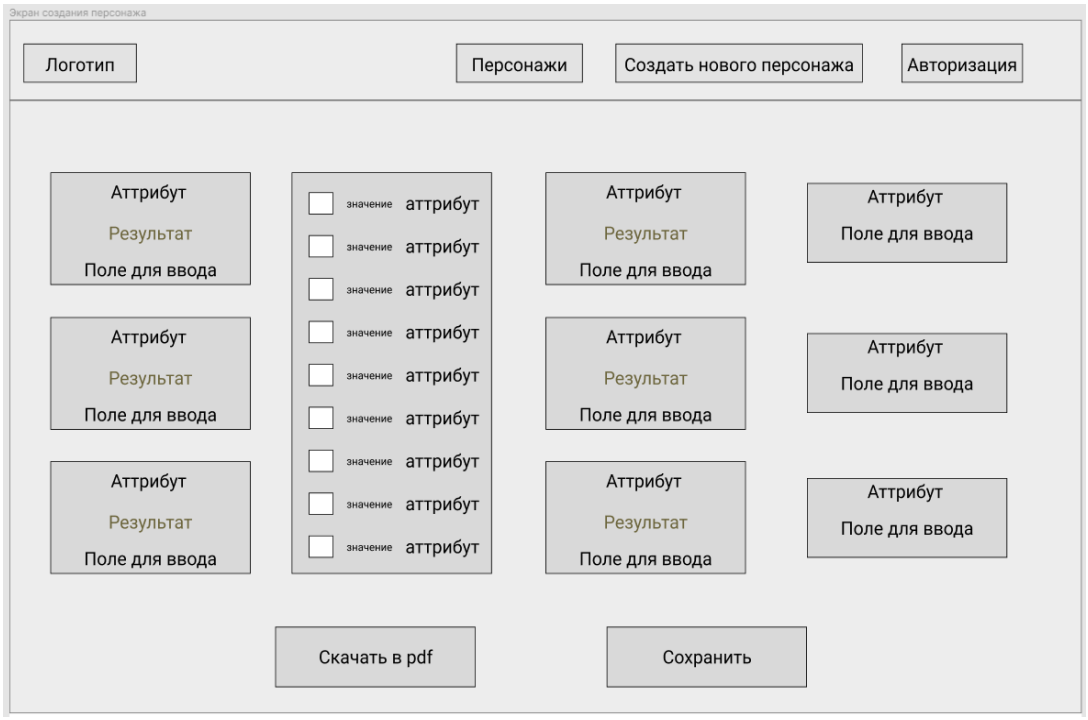

Рис. 9. Макет страницы создания персонажа

Параметры некоторых характеристик могу зависеть от данных, введённых данных в других полях. Например, значение полей в блоке «Испытания», которое состоит из поля с отметкой, поля заполнения данных и текстового поля. Данный блок зависит от базовых атрибутов (сила, ловкость и т.д.), которые влияют на данную характеристику, и зависит от значения в блоке «Бонуса умения», который состоит из поля заполнения данных и текстового поля.

Когда пользователь заполнит все данные, он может нажать на кнопку «скачать в pdf» и получить лист персонажа, сохраненным в файле формата pdf. Авторизованный пользователь имеет возможность просмотра сохраненных персонаже, листы которых сохраняются при нажатии на кнопку сохранить. При нажатии на кнопку «Персонажи» авторизованный пользователь перейдет на страницу просмотра сохраненных персонажей, на которой пользователь увидит список персонажей, представленный в виде списка карточек с именем персонажа, номером в списке и кнопкой «Открыть». При нажатии на кнопку «Открыть» пользователь будет перемещен на

страницу редактирования персонажа, соответствующего карточке сохранения, на которой располагается кнопка. На рисунке 10 представлен макет страницы сохраненных персонажей.

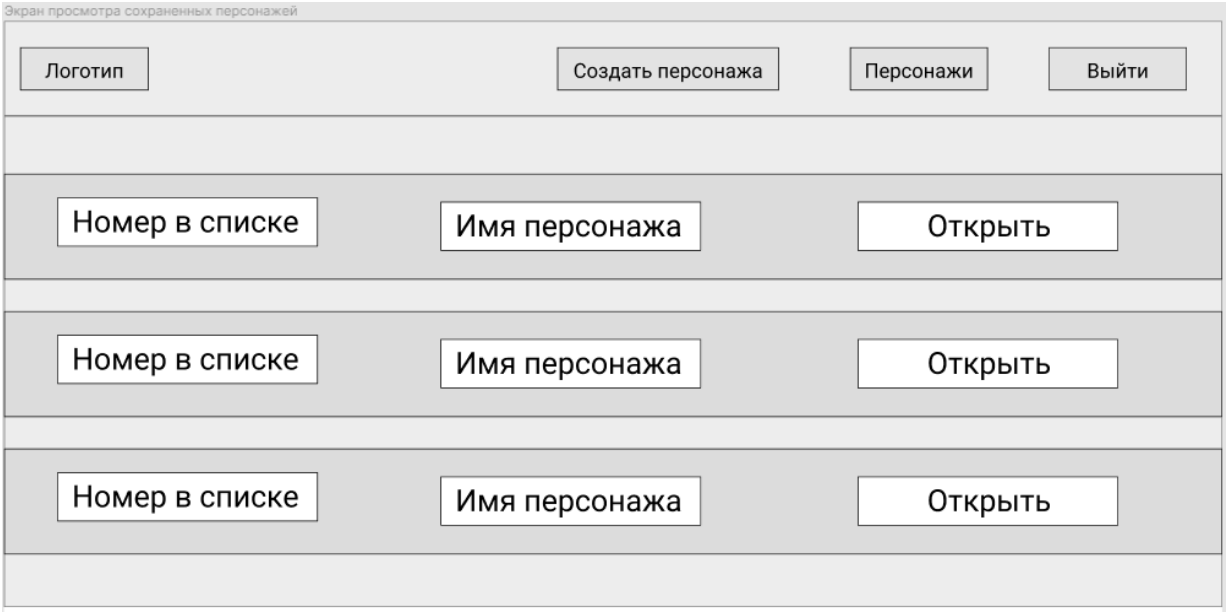

Рис. 10. Макет страницы просмотра сохраненных персонажей

3. РЕАЛИЗАЦИЯ СИСТЕМЫ

3.1. Инструменты реализации

Серверная часть системы реализована на высокоуровневом языке программирования JavaScript версии ES6 с использованием платформы Node.js. JavaScript – это мультипарадигменный язык программирования, обычно используется для программного доступа к объектам приложений. Наиболее широко применяется в браузерах. Node.js – опенсорсная кроссплатформленная среда выполнения для JavaScript, построенная на базе ядра V8 от Google, который используется в браузере Google Chrome. Асинхронные механизмы позволяют серверу одновременно обрабатывать тысячи подключений, не требуя от программиста управлять потоками и организовывать параллельное выполнение кода. Так же платформа предоставляет разработчику неблокирующие базовые механизмы ввода и вывода.

На сервере данные хранятся в базе данных, созданной с помощью документированной СУБД MongoDB [10], которая была выбрана по причине свободного распространения и возможности интегрирования с платформой Node.js.

Клиентская часть системы реализована для веб-браузеров на высокоуровневом языке программирования JavaScript, с использованием библиотеки React.js. Она позволяет вам собирать сложный UI из маленьких изолированных кусочков кода, называемых «компонентами», используя virtual DOM[16]. Для локального хранения данных, используются внутренние состояния (state).

3.2. Реализация сервера

Работа с базой данных

Работа с базой данных осуществляется при помощи пакета mongoose. Для создания документа в базе данных требуется модель данных, называемая «scheme». Пример модели данных показан на рисунке 11.

```
const schema = new Schema({
    email:{type:String, required: true, unique: true},
    password: {type:String, required:true},
    heroes: {type: Types.ObjectId, ref: 'Hero'}
});
```

Рис. 11. Модель данных «User»

Схема описывает хранения данных в формате ключ-значение. У каждого поля есть обязательный параметр «type». Основные типы данных, которые использовались в данном проекте: string (строка), Types.ObjectId (идентификатор документа на который ссылается текущий документ и Object. Некоторые поля имеют и необязательные параметры «required» или «unique».

Безопасность данных

Данные хранятся в виде результата вычисления криптографической хеш-функции bcrypt от пароля. bcrypt – адаптивная криптографическая функция формирования ключа, используемая для защищенного хранения паролей [43].

В случае успешной аутентификации клиента, сервер высылает приложению сгенерированный JWT, используемый в дальнейшем для авторизации клиента. JWT (JSON Web Token) – это открытый стандарт для обеспечения безопасности передачи пакетов между сторонами [42]. Токен передается в зашифрованном виде и состоит из трех частей, разделенных точками.

1. Закодированный заголовок, содержащий информацию о типе стандарта («JWT») и об алгоритме шифрования (например, «HS256»).

2. Тело токена, содержащее метаданные. В веб-приложении «D&D5 Hero» токен содержит идентификатор пользователя и дату истечения токена, после которой токен будет считаться недействительным.

3. Цифровая подпись, которая содержит ключ для шифрования всех частей токена.

3.3.Маршрутизация

Взаимодействие клиента и сервера реализовано по средствам HTTP-запросов от клиента к серверу. Пример адресов в приложении представлен в таблице 1. После отправки запроса от клиента к серверу, сервер определяет обработчик для текущего адреса, затем обрабатывает информацию в блоке try-catch, для обработки возможных ошибок.

Табл. 1. REST запросы

Адрес запроса	Описание	Метод
/api/auth/register	регистрация пользователя	POST
/api/auth/login	авторизация пользователя	POST
/api/hero/save	сохранение персонажа	POST
/api/hero/:id	сохраненный персонаж	GET
/api/heroes/	список сохраненных персонажей	GET
/api/hero/create-hero	создать персонажа	POST
/api/hero/get-hero	получить персонажа	GET

Для примера, мы рассмотрим обработку POST запроса по адресу /api/auth/login. После отправки POST запроса по данному URL начинается обработка данных в блоке try-catch, для обработки возможной ошибки. Код обработчика приведен на рисунке 12. В секции try из тела запроса собираются данные полей email и password, затем сервер обращается к базе данных, чтобы найти пользователя с таким же email-ом. Если пользователь с такими данными не был найден, сервер отправляет статус 400 (неверный запрос) и сообщение: «Такой пользователь не найден». В случае, если пользователь с такими данными найден, при помощи библиотеки bcrypt[1] сравниваются хэш пароли из базы данных и тела запроса. Если пароли не совпадают, сервер снова отправляет запрос со статусом 400 и сообщение «Неверные введенные данные».

```
router.post('/login', async (req, res) => {
try {
      const {email, password} = req.body;
      const userLogin = await User.findOne({email});
if(!userLogin){
      res.status(400).json({message:'Такой пользователь не найден'});}
const isMatch = await bcrypt.compare(password, userLogin.password)
if(!isMatch){
      res.status(400).json({message:'Неверные введенные данные'});}
const token = jwt.sign(
      {userId: userLogin.id},
      config.get('jwtKey'),
      {expiresIn: '1h'});
res.json({token, userId: userLogin.id});
}
catch(e) {
      res.status(500).json({message:'что-то пошло не так'});
}})
```

Рис. 12. Обработка авторизации на сервере

В целях защиты данных, пользователю не сообщается, в чем конкретно было допущены ошибка. После успешной проверки паролей на клиент будет отправлен jwt.

Если необходимые данные не были получены, тогда создается ошибка и передается в блок catch, после чего сервер отправляет на клиент статус 500 (внутренняя ошибка сервера) и сообщение об ошибке в теле ответа.

3.4.Реализация клиента

Для реализации клиента использовалась библиотека React.js[15]. Клиент состоит из множества компонентов и страниц. Страницы являются базовым шаблоном, который включает в себя компоненты. На пример, страница «CreateHero» содержит в себе состояния всех полей, для того, чтобы при изменении поля происходила повторная отрисовка компонента с актуальной информацией. Пример состояния компонента приведен на рисунке 13.

```
    this.state = {code: "01",
      baseAttr: {
        strength: 0,
        agility: 0,
        stamina: 0,
        intellig: 0,
        wise: 0,
        charisma: 0,
        bonusSkill: 0,
      },
```

Рис. 13. Состояния страницы «CreateHero»

Также «CreateHero» содержит в себе обработчики событий, которые передает дальше в дочерние компоненты. При возникновении события «onChange» (событие возникает при изменении значения полей), вызывается соответствующий обработчик. На рисунке 14 приведен пример обработчика.

```
 handleAttr = (e) => {
    const eventValue = e.target.value;
    const eventName = e.target.name;
    if (eventName === "bonusSkill") {      this.setState((state) => ({
baseAttr:{ ...state.baseAttr, ...{ [eventName]: Number(eventValue) } },
value: { ...state.value, ...{ [eventName]: Number(eventValue) } },}));}
else {if (Number(eventValue) > 30) {this.setState((state) => ({
      baseAttr: { ...state.baseAttr, ...{ [eventName]: staticX[30] } },
         value: { ...state.value, ...{ [eventName]: eventValue } },}));
      } else {this.setState((state) => ({
   baseAttr:{ ...state.baseAttr,...{ [eventName]: staticX[eventValue] },},
          value: { ...state.value, ...{ [eventName]: eventValue } },
        }));}
```

Рис. 14. Функция обработчик «handleAttr»

После того, как произошло событие, в качестве аргумента обработчик принимает объект события (change). Затем идет хеширование параметров «name» и «value», полученных из объекта события.

Страница «CreateHero» содержит в себе множество компонентов, в которые передает требуемые состояния и обработчики, компоненты в свою

очередь обрабатывают переданные в них свойства и отрисовывают интерфейс. Рассмотрим выше сказанное на примере компонента «BaseAttr». Компонент возвращает JSX[9], содержащий стандартные HTML тэги, с атрибутами, значения, переданные от компонента родителя и обработчики. Реализация компонента представлена на рисунке 15.

```
const BaseAttr = (props)=>  {
return (
<div className="col s4 base_attr_wraper">
<div className="input-field col s12 hero_data_attr">
<span className="base_attr_card_title">Сила:</span>
<span className="res_value" data-name="strength">
{props.baseData.strength}</span>
<input id="hero_strength"
type="text" className="validate" name="strength"
value={props.value.strength} onChange={props.handleAttr.bind(this)}/>
</div>
<div className="input-field col s12 hero_data_attr">
<span className="base_attr_card_title">Ловкость:</span>
<span className="res_value" data-name="agility">
{props.baseData.agility}</span>
<input id="hero_agility" type="text" className="validate"
name="agility" value={props.value.agility}
onChange={props.handleAttr.bind(this)}/>
</div>
</div>
)
}export default BaseAttr
```

Рис. 15. Компонент «BaseAttr»

В клиенте используются, как классовые, так и функциональные компоненты. Для примера, возьмем компонент «SaveButtons». Реализация компонента представлена на рисунке 16. Это небольшой компонент для сохранения персонажа на аккаунт. Он принимает данные из родительского компонента и токен из контекста. Контекст позволяет передавать данные через дерево компонентов без необходимости передавать данные на промежуточных уровнях. Также импортируется пользовательский *хук,* который осуществляет запрос на сервер. Хуки - это функции, с помощью кото-

рых можно «подцепиться» к состоянию и методам жизненного цикла React из функциональных компонентов. После того, как пользователь нажимает на кнопку «Сохранить», функция «saveHero» делает POST запрос, по адресу /api/hero/save, с телом, в котором хранятся данные о персонаже, переданные их родительского компонента. Результат запроса выводится пользователю на экране.

```
import React, { useContext } from "react";
import M from "materialize-css";
import { useHttp } from "../hooks/http.hook";
import { AuthContext } from "../context/AuthContext";
const SaveButton = (props) => {
const { request } = useHttp();
const auth = useContext(AuthContext);
const dataHero = props.dataHero;
const saveHero = async (e) => {
e.preventDefault();
try {const reqData = await request("/api/hero/save","POST",{ dataHero },
{Autorization: `Bearer ${auth.token}`,});
M.toast({ html: `${reqData.message}` });
} catch (err) {
M.toast({
html: `${err.message}` || "При отправке запроса что-то пошло нет так",});
}};
return (
<>
<button
className=' col s6 btn pink lighten-1 btn'
onClick={(e) => saveHero(e)}>Сохранить</button>
</>
);};
export default SaveButton;
```

Рис. 16. Реализация компонента «SaveButton»

3.5.Пользовательский интерфейс

Для простоты интерфейса и разметки использовалась библиотека materialize.css [11], имеющая готовые компоненты и удобную сетку для позиционирования элементов. Все элементы размещены на одной странице,

чтобы сделать навигацию удобной. Интерфейс страницы создания персонажа продемонстрирован на рисунке 17.

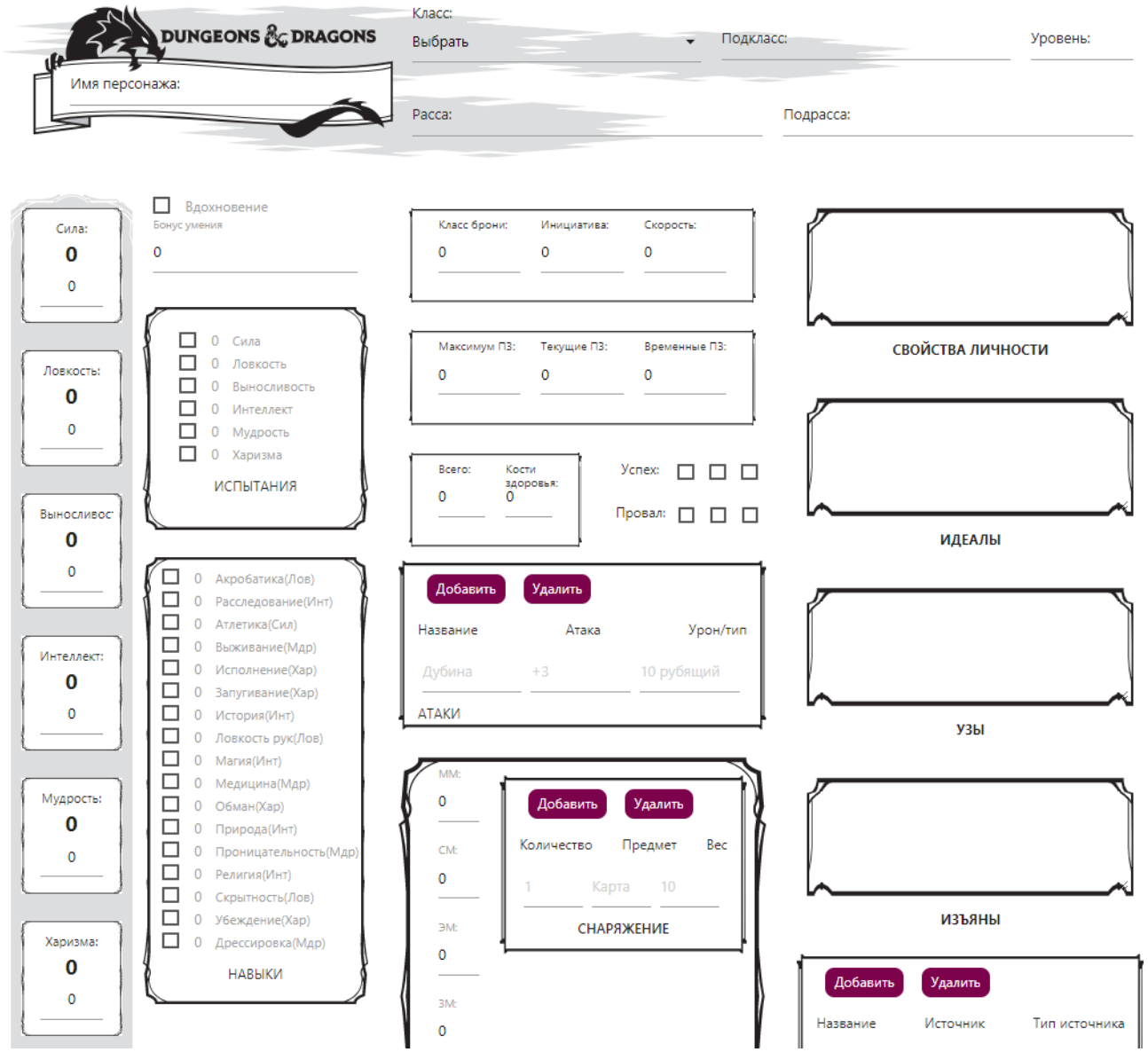

Рис. 17. Страница создания персонажа

Блоки, содержащиеся в интерфейсе можно разделить по нескольким типам: простой текстовый блок, текстовый блок с дополнительными функциями и блок с отметками.

Простой текстовый блок содержит в себе название (title), поле для ввода информации. Некоторые простые блоки могут содержать дополнительное место для отображения рассчитанного значения. Пример простого текстового блока продемонстрирован на рисунке 18.

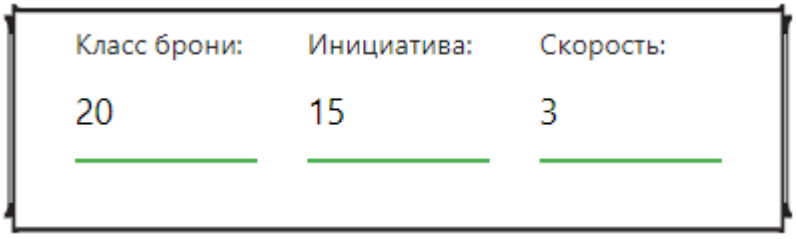

Рис. 18. Интерфейс простого текстового блока

Блоки с функцией «добавить поле» имеют дополнительные кнопки «add» и «del». При нажатии на кнопку «add» будет добавлена группа полей в конец списка, при нажатии кнопки «del» группа полей будет удалена из начала списка. Пример текстового блока с дополнительными функциями продемонстрирован на рисунке 19.

Рис. 19. Интерфейс текстового блока с дополнительными возможностями

Блок «испытания» реализован как блок с отметками, в котором представлены секциями, содержащие поле типа «checkbox», поле для вывода рассчитанного значения и название атрибута. Если пользователь «выбирает» атрибут, то конечно значение будет высчитываться на основе базового показателя, плюс значение из поля «бонус умения». В случае, если пользователь не «выбирает» атрибут, то значение будет равняться базовому показателю. На рисунке 20 приведен пример заполнения блока «испытания», где ко всем значениям атрибутов, отмеченных галочкой, добавляется зна-

чение поля «бонус умения», которое располагается над блоком «испытания».

Рис. 20. Интерфейс блока с отметками

Страницы для регистрации и авторизации, были объединены в одну, чтобы не заставлять пользователя переходить по множеству страниц. Форма имеет поля «email» и «пароль». Данные, введенные в эти поля должны пройти первичную проверку (минимальная длинна пароля, наличие символа «@» в поле «email»), если они не соответствуют, то пользователю будет выведено сообщение о том, что данные не прошли проверку. Если пользователь уже имеет аккаунт и хочет авторизоваться в приложении, ему нужно нажать кнопку «Войти», если данные будут верны, он будет перенаправлен на страницу создания персонажа. Для регистрации нужно нажать кнопку «Регистрация», если данные проходят проверку. Интерфейс формы авторизации представлен на рисунке 21.

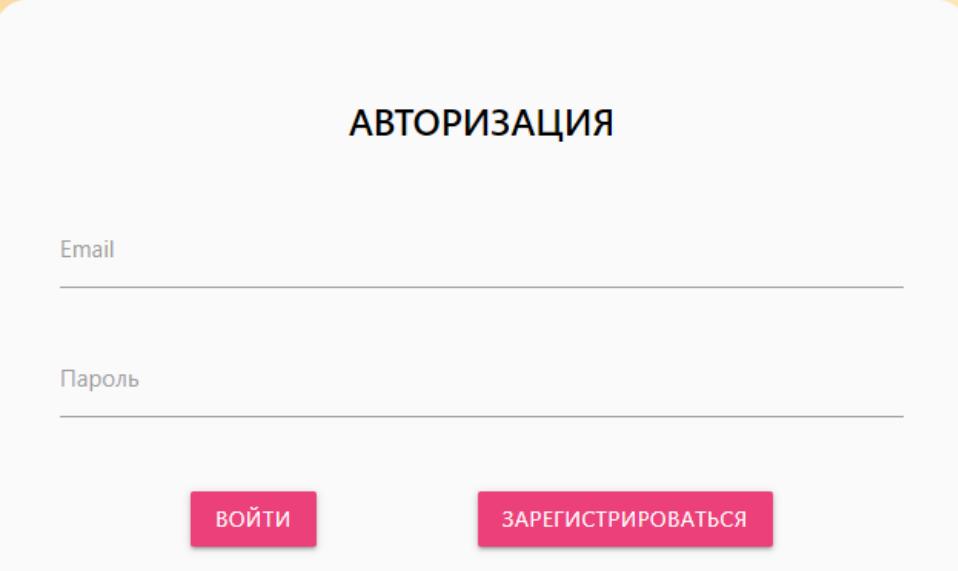

Рис. 21. Форма авторизации

На странице «Персонажи» пользователю предоставляется список со всеми сохраненными персонажами, с возможностью посмотреть имя персонажа и перейти к его редактированию. Каждый элемент в списке содержит несколько полей: номер в списке, имя сохраненного персонажа, и кнопка, чтобы перейти к редактированию персонажа. Интерфейс списка персонажей продемонстрирован на рисунке 22.

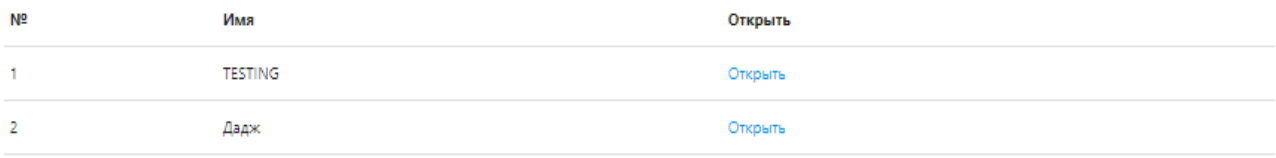

Рис. 22. Список сохраненных персонажей

4. ТЕСТИРОВАНИЕ СИСТЕМЫ

Для тестирования системы применялось функциональное тестирование, т.е. тестирование программного обеспечения в целях проверки реализуемости функциональных требований.

Табл. 2. Тестирование приложения

№	Аспект работы	Действия	Результат	Тест пройден?
1	Регистрация	1) На странице авторизации ввести email и пароль. 2) Нажать кнопку «регистрация» 3) Получить оповещение о результате запроса	Выводится сообщение об успешной регистрации.	Да
2	Авторизация	1) На странице авторизации ввести email и пароль. 2) Нажать кнопку «Войти». 3) Получить перенаправление на страницу создания персонажа.	При успешной авторизации произошло перенаправление на страницу создания персонажа	Да
3	Сохранение персонажа	1) На странице создания персонажа заполнить поля. 2) Нажать кнопку «сохранить». 3) Получить оповещение о результате запроса.	После нажатия на кнопку «сохранить» было получено сообщение о сохранение данных.	Да
4	Просмотр сохраненных персонаже	1) В шапке сайта нажать на кнопку «Персонажи». 2) Получить перенаправление по адресу: /vault/: id. 3) Получить список персонажей	После нажатия на кнопку «Персонажи», произошло перенаправление по адресу: /vault/: id. Затем получен список персонажей.	Да

Окончание табл. 2

№	Аспект работы	Действия	Результат	Тест прой ден?
5	Редактирование сохраненных персонажей	1) На странице сохранённых персонажей нажать на кнопку «открыть». 2) Получить перенаправление по адресу: /create/: id. 3) Получить интерфейс с заполненными полями, в соответствии с данными	Произошло перенаправление по адресу: /create/: id, где было выведен интерфейс для создания персонажа с заполненными ранее полями	Да
6	Получение персонажа в pdf.	1) На странице создания персонажа нажать кнопку «скачать» 2) Получить pdf файл с информацией о персонаже.	После нажатия на кнопку, был получен файл pdf, в которым была информация о созданном персонаже	Да

Табл. 3. Тестирование сервера

№	Аспект работы	Действия	Результат	Тест прой ден?
1	Сохранить своего персонажа	1) Оправить POST запрос на соответствующий URL, содержащий заголовок с JWT. Тело запроса JSON, с данными для сохранения.	В базе данных изменяются отправленные параметры. Получен JSON с полем message и значением «данные сохранены»	Да
2	Изменить своего персонажа	1) Отправить POST запрос, на соответствующий URL, содержащий заголовок с JWT. Тело запроса JSON, с параметрами персонажа и его идентификатора.	В базе данных обновляется соответствующий документ. Получен JSON с полем message»	Да

Окончание табл. 3

№	Аспект работы	Действия	Результат	Тест прой ден?
3	Получить персонажа в pdf	1) Отправить POST запрос, на соответствующий URL, содержащий заголовок с JWT. Тело запроса JSON, с параметрами персонажа.	Данные обработаны и отправлены на клиент в html формате.	Да

Табл. 4. Тестирование интерфейса конструктора

№	Элемент интерфейса	Действия	Результат	Тест прой ден?
1	Текстовое поле	1) Ввести данные в текстовое поле. 2) Смена цвета поля, смена позиция «label»	После клика на поле, появился курсор. После ввода данных цвет поля сменился и название переместилось наверх.	Да
2	Список с выбором	1) Кликнуть на поле со списком. 2) Увидеть список с возможными вариантами. 3) Выбрать один из вариантов. После чего пустое поле заменяется на выбранный вариант.	Кликнув по полю «класс» был получен список с возможными вариантами. Выбранное значение заменило пустое поле.	Да
3	Текстовое поле с рассчитываемым результатом	1) Ввести данные в поле. 2) Получить рассчитанные данные в специальном блоке над текущим полем.	После ввода данных, число над полем изменилось.	Да

Окончание табл. 4

№	Элемент интерфейса	Действия	Результат	Тест прой ден?
4	Блок содержащий поля с выбором	1) Выбрать атрибут в блоке. 2) Значения поля поменялось на «checked» и изменилось изображение. 3) Получить рассчитанный результат (на основе поля «бонус умения» в соседнем поле).	После выбора атрибута, было получено измененное значение в соседнем поле.	Да
5	Кнопка для добавления секции полей	1) Нажать на кнопку «Добавить». 2) Добавление новой секции полей в текущем блоке.	После нажатия на кнопку «Добавить» была добавлена секция полей.	Да
6	Кнопка для удаления секции полей	1) Нажать на кнопку «Удалить». 2) Удаление одной секции полей в текущем блоке.	После нажатия на кнопку «Удалить» была удалена секция полей.	Да

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы бакалавра было разработано веб-приложение «D&D5 Hero» для создания персонажей DnD игр, которое можно использовать для создания листов персонажей для игры Dragons and dungeons версии 5-ой редакции. Код системы составил свыше 1000 строк кода на языке javascript.

Основные результаты

1. Выполнен анализ предметной области и произведен обзор существующих решений.
2. Спроектирован и реализован сервер системы.
3. Спроектирован и реализован клиент.
4. Проведено тестирование приложения.

Был реализован конструктор для создания персонажа DnD игр. Была реализована возможность получить созданный лист персонажа в pdf формате. Также в разработанном веб-приложении можно сохранять созданных персонажей и редактировать их в случае необходимости.

Направление дальнейших исследований

Дальнейшая работа над приложением будет направлена на улучшение дизайна интерфейса, на добавление возможности редактирования личных данных, на улучшение листа персонажа pdf файла и на добавление поддержки разных версий DnD игр.

ЛИТЕРАТУРА

1. bcrypt [Электронный ресурс] URL: https://www.npmjs.com/package/bcrypt (дата обращения 15.05.2020).
2. CharacterBuild-er.[Электронный ресурс]URL:https://www.aidedd.org/dnd-builder/index.php (дата обращения: 14.05.2020).
3. CSSSpecification. [Электронный ресурс] URL: http://www.w3.org/Style/CSS/current-work (дата обращения: 14.05.2019).
4. D&D Beyond. [Электронный ресурс] URL:https://www.dndbeyond.com/marketplace/adventures/lost-mine-of-phandelver (дата обращения: 13.05.2020).
5. D&D book. [Электронный ресурс] URL: https://www.dndbeyond.com/profile/Why_I_must/characters/28391338/builder#/description/manage (дата обращения: 14.05.2020).
6. Geekster. [Электронный ресурс] URL: https://geekster.ru/hot-news/5-izmenenij-v-dungeons-dragons-5/ (дата обращения: 13.05.2020).
7. HTML/Specification. [Электронный ресурс] URL: http://www.w3.org/community/webed/wiki/HTML/Specifications#Specifications (дата обращения: 14.05.2019).
8. JSON Web Tokens. [Электронный ресурс] URL: https://jwt.io/ (дата обращения: 15.05.2020)
9. JSX. [Электронный ресурс] URL:https://ru.reactjs.org/docs/introducing-jsx.html (дата обращения: 13.05.2020).
10. MongoDB [Электронный ресурс] URL: https://www.mongodb.com/ (дата обращения 15.05.2020).
11. Materialize. [Электронный ресурс] URL: https://materializecss.com/color.html (дата обращения: 14.05.2020).
12. Node.js [Электронный ресурс] URL: https://nodejs.org/ru/docs/ (дата обращения 15.05.2020).

13. Provos N., Mazieres D. A Future-Adaptable Password Scheme. // USENIX Annual Technical Conference, FREENIX Track, 1999. – С. 81–91.

14. Roll20. [Электронный ресурс] URL: https://roll20.net/ (дата обращения: 14.05.2020).

15. React. [Электронный ресурс] URL: https://reactjs.org/docs/getting-started.html (дата обращения: 13.05.2019).

16. VirtualDom. [Электронный ресурс] URL: https://learn-reactjs.ru/faq/virtual-dom-and-internals (дата обращения: 13.05.2019).